z/OS V2R3 zFS File System Encryption and Compression

Beginning in z/OS V2R3, new and existing zFS file system data can be encrypted and compressed, using the DFSMS access method encryption and the zEDC compression method. After a file system is encrypted, compressed, or both, additional new entries will also be encrypted, compressed, or both.

The dataset used in the format of the zFS file system is protected if associated with an encryption key label to identify an encryption key. The dataset itself cannot be opened, read, or modified by a user with access or visibility to the dataset without being authorized to the associated encryption key.

To prevent potential unauthorized permitting of access to the data in mounted file systems with a key label, the formatted zFS file system datasets with a key label cannot move ownership to another system and be mounted without the userids for OMVS and zFS being authorized to the associated encryption keys. Depending on if zFS is in a colony (outside of OMVS and in its own address space) or in OMVS (resides within the OMVS address space), the OMVS (e.g. OMVSKERN) and zFS (e.g. DFS) userids must be trusted, assigned the appropriate attributes, or permitted to the proper RACF resources, and authorized to the encryption key(s) associated with the file system datasets. See the Distributed File Service zFS Administration manual for more information. OMVS and zFS will then have authority to mount and have access to the file system dataset with the key label. While the encryption eligible file system is mounted with the proper access authority and encryption key access, the access control to the data itself is controlled by the UNIX protection methods. The encryption and decryption will take place with the access to the file data. After the encryption eligible zFS formatted file system dataset is mounted and the user can access the files and directories via UNIX access methods (e.g. permissions, ACLs, security labels, FSACCESS, etc), the data can be accessed even though the individual user is not permitted to the encryption key.

A new file system can be defined and formatted to allow any data added to be automatically encrypted, compressed, or both. Use of the new variables, *format_encryption=on* or *format_compression=on*, can be set in your IOEFSPRM configuration file if you want data in all new zFS file systems to be automatically encrypted, compressed, or both. The default for both is off. This global encryption and compression default can be overridden by specifying the *-noencrypt*, *-nocompress*, *-encrypt*, or *-compress* keyword on the zFS format. If the default is set to *on* for the *format_encryption* variable (*format_encryption=on*), the dataset will need to be defined with a key label, which should exist in ICSF.

Version 1.4 zFS aggregates cannot be encrypted. Consider defining new file systems with the DFSMS extended format option and the new DFSMS DEFINE CLUSTER *ZFS* keyword. Use of the *ZFS* keyword instead of LINEAR will allow key labels to be assigned to any VSAM linear data set that is supported by zFS. For encryption of a zFS file system, an associated key label needs to be defined and assigned to the dataset. The *-keylabel* option on the *zfsadm define*, the *zfsadm encrypt*, or other appropriate commands is only needed when a zFS aggregate is encrypted for the first time and if it was not specified when the VSAM linear (ZFS) data set was created.

Existing zFS file system data can be encrypted and compressed using the new zFS zfsadm conversion commands, *zfsadm encrypt* and *zfsadm compress*. If the compression is performed first the amount of data to be encrypted is smaller which may improve performance. There are also new zFS zfsadm commands which will allow decryption (*zfsadm decrypt*) and decompression (*zfsadm decompress*). Encrypting or compressing an existing file system is a long-running administrative command, which will usually continue to execute on the zFS owning system in background tasks until complete or interrupted, even if the starting session is detached. The long-running command thread pool must have an available foreground thread on the zFS owning system. See the IOEFSPRM configuration option *long_cmd_threads* for more information. You can cancel or interrupt the conversion for encrypt, decrypt, compression, or decompression execution with the appropriate *zfsadm -cancel* option, *unmount with the force option*, or *during shutdown*, which will stop the operation at the percent of completion. The execution can be resumed or initiated by issuing the appropriate command for the partial encryption (-encrypt or -decrypt) or compression (-compress or -decompress). The progress of the operation can be monitored with *fsinfo*. During this process, background tasks on the zFS owning system will process the objects in the file system. Encryption will occur for all security information, access control lists, symbolic link contents, and file contents. Application access is fully allowed to the file system during the operation. A file system that is in a partially encrypted or decrypted state cannot be compressed. Likewise, a file system that is in a partially compressed or decompressed state cannot be encrypted. The *zfsadm fsinfo* or the console '*modify omvs,pfs=zfs,fsinfo…*' commands can be used to monitor the percent of completion.

The decrypt process does not remove key labels. File systems that have key labels assigned cannot be mounted on a release prior to z/OS V2R3, even if those file systems have not been encrypted or are currently not encrypted. Therefore, if there is no zFS system in the shared file system environment that is eligible to own a file system with a key label assigned to it, the file system will be inaccessible.

Only files that are larger than 8K will be compressed. To improve performance of the compression I/O, consider specifying the *edcfixed* option in the IOEFSPRM parameter *user_cache_size*. See the IOEFSPRM configuration option *user_cache_size* for more information. If the user cache is not registered with the zEDC Express service, zFS will attempt to register the user cache after the *zfsadm compress* command completes. There is also a ZFS_VERIFY_COMPRESSION_HEALTH health checker to provide information on the zFS configuration relative to zEDC support.

Encryption and compression for a zFS file system is not supported before z/OS V2R3. All systems in a sysplex must be at least z/OS V2R3 before encryption and compression can begin. Do not begin the encryption or compression process until you know that no system will be regressed to an earlier release. ICSF must be available, configured, and active for encryption. Consider starting ICSF early in the IPL process and with SUB=MSTR to allow starting before JES (example: S CSF,SUB=MSTR). We have it issued early in the COMMNDxx member. zEDC must be available, configured, and active for compression to occur on the system. To obtain the full benefit of encryption and compression together, consider both for eligible zFS file systems.

Some Tests Performed.

The following are some tests we performed in an all z/OS V2R3 sysplex. We have some of our systems executing zFS in the OMVS address space and other systems where zFS is executing in a colony address space. Both userids for OMVS and zFS have authorization to the VSAM datasets and the encryption key. The *format_encryption* and *format_compression* configuration variables are defaulted as *off* so we can control which file systems are formatted with encryption and compression. Commands and displays were issued from a superuser.

On each system in the V2R3 sysplex, we see in the ZFS_VERIFY_COMPRESSION_HEALTH health check that the systems are appropriately configured for zFS compression.
Excerpt from the health check:

```
...
An exception condition has not been detected. Relative to zEDC
support, zFS is appropriately configured.
...
```

Define a new zFS file system for Encryption and Compression. Using IDCAMS:

Using IDCAMS and the ZFS parameter and the KEYLABEL parameter, we allocated an extended format VSAM dataset.
Note: DATACLAS(SMSOE) has the extended format definitions.
Excerpts from JCL and job output:

```
…
//DEFINE  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//AMSDUMP  DD SYSOUT=*
//SYSIN    DD *
  DEFINE CLUSTER (NAME(OMVSSPT.ZFSEC.ZFS) -
  ZFS    CYL(10 10) SHAREOPTIONS(3) -
  KEYLABEL(ZFS.AES.SECURE.KEY) -
  STORCLAS(SMSOE) -
  DATACLAS(SMSOE) -
  MGMTCLAS(SMSOE))
  LISTCAT ENTRIES(OMVSSPT.ZFSEC.ZFS) -
  ALL
/*
…
IGD17150I DATA SET OMVSSPT.ZFSEC.ZFS IS
ELIGIBLE FOR ACCESS METHOD ENCRYPTION. KEY LABEL IS
(ZFS.AES.SECURE.KEY)
IGD17172I DATA SET OMVSSPT.ZFSEC.ZFS
IS ELIGIBLE FOR EXTENDED ADDRESSABILITY
IDC0512I NAME GENERATED-(D) OMVSSPT.ZFSEC.ZFS.DATA
IDC0181I STORAGECLASS USED IS SMSOE
IDC0181I MANAGEMENTCLASS USED IS SMSOE
IDC0181I DATACLASS USED IS SMSOE
```

```
IDCAMS  SYSTEM SERVICES
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
…
```

Looking at an IDCAMS LISTC (`LISTCAT ENTRIES(OMVSSPT.ZFSEC.ZFS) ALL`) listing from a z/OS V2R3 system, we see the **ZFS**, **Extended Format**, and the **KEYLABEL** attributes associated with the dataset.

```
…
LISTCAT ENTRIES(OMVSSPT.ZFSEC.ZFS) -
ALL
CLUSTER ------- OMVSSPT.ZFSEC.ZFS
…
ENCRYPTIONDATA
DATA SET ENCRYPTION----(YES)
DATA SET KEY LABEL-----ZFS.AES.SECURE.KEY
…
SHROPTNS(3,3)   RECOVERY     UNIQUE           NOERASE     LINEAR
NONSPANNED      EXTENDED     EXT-ADDR             ZFS
…
```

We then used the ZFS IOEFSUTL utility to format the file system as a version 1.5 file system with encryption and compression eligible. This will allow data in the mounted file system to be encrypted and compressed. We verified the format with *zfsadm fsinfo* before mounting. Excerpts from JCL and job output:

```
…
//FORMAT  EXEC PGM=IOEFSUTL,REGION=0M,
// PARM=('format -aggregate OMVSSPT.ZFSEC.ZFS -version5 -encrypt
//             -compress')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//STDOUT   DD SYSOUT=*
//STDERR   DD SYSOUT=*
//CEEDUMP  DD SYSOUT=*
…
IOEZ00004I Formatting to 8K block number 900 for primary extent of
OMVSSPT.ZFSEC.ZFS.
IOEZ00005I Primary extent loaded successfully for OMVSSPT.ZFSEC.ZFS.
IOEZ00077I HFS-compatibility aggregate OMVSSPT.ZFSEC.ZFS has been
successfully created
…
```

Using the *zfsadm fsinfo* display we can see that the file system is now at version 1.5 and encryption and compression eligible.

/bin/zfsadm fsinfo OMVSSPT.ZFSEC.ZFS

```
File System Name: OMVSSPT.ZFSEC.ZFS

*** owner information ***
Owner:               n/a             Converttov5:            OFF,n/a
Size:                7200K           Free 8K Blocks:         881
Free 1K Fragments:   0               Log File Size:          120K
Bitmap Size:         8K              Anode Table Size:       8K
```

```
File System Objects: 3              Version:                1.5
Overflow Pages:      0              Overflow HighWater:     0
Thrashing Objects:   0              Thrashing Resolution:   0
Token Revocations:   0              Revocation Wait Time:   0.000
Devno:               0              Space Monitoring:       0,0
Quiescing System:    n/a            Quiescing Job Name:     n/a
Quiescor ASID:       n/a            File System Grow:       OFF,0
Status:              NM,EN,CO
Audit Fid:           D7F2E4E2 F9F50332 0000
Backups:             0              Backup File Space:      0K

File System Creation Time: May 10 14:56:24 2018
Time of Ownership:         May 10 15:02:40 2018
Statistics Reset Time:     May 10 15:02:40 2018
Quiesce Time:              n/a
Last Grow Time:            n/a

Connected Clients:   n/a


Legend: NM=Not mounted, EN=Encrypted, CO=Compressed
```

When we tried to copy the dataset from a user id that was not authorized to the encryption key, we received the following for insufficient access authority.

```
…
//REPRO     EXEC PGM=IDCAMS
//SYSPRINT  DD   SYSOUT=*
//DDI       DD   DISP=SHR,DSN=OMVSSPT.ZFSEC.ZFS
//DDO       DD   DISP=SHR,DSN=OMVSSPT.ZFSEC.ZFS.REPRO
//SYSIN     DD   *
   REPRO -
      INFILE(DDI) -
      OUTFILE(DDI)
/*
…
ICH408I USER(U050001 ) GROUP(TESTJ00 ) NAME(USSTEAM)
  ZFS.AES.SECURE.KEY CL(CSFKEYS )
  INSUFFICIENT ACCESS AUTHORITY
  ACCESS INTENT(READ    )  ACCESS ALLOWED(NONE    )
```

Define a new zFS file system for Encryption and Compression.

Using zfsadm define and zfsadm format:

Using the *zfsadm define* and *zfsadm format* commands, we allocated and formatted a Non-Extended format VSAM dataset, providing the key label

/bin/zfsadm define -aggregate OMVSSPT.ZFSEC.NONEXT.ZFS -storageclass SMSOE -managementclass SMSOE -cylinders 10 10 -keylabel ZFS.AES.SECURE.KEY

```
IOEZ00248I VSAM linear dataset OMVSSPT.ZFSEC.NONEXT.ZFS successfully created.
```

/bin/zfsadm format -aggregate OMVSSPT.ZFSEC.NONEXT.ZFS -version5 -encrypt -compress

```
IOEZ00077I HFS-compatibility aggregate OMVSSPT.ZFSEC.NONEXT.ZFS has been
successfully created
```

Using *zfsadm fsinfo*, we checked that the newly formatted file system is encryption and compression eligible.

/bin/zfsadm fsinfo -aggregate OMVSSPT.ZFSEC.NONEXT.ZFS

```
File System Name: OMVSSPT.ZFSEC.NONEXT.ZFS

Owner:                n/a             Converttov5:            OFF,n/a
Size:                 7200K           Free 8K Blocks:         881
Free 1K Fragments:    0               Log File Size:          120K
Bitmap Size:          8K              Anode Table Size:       8K
File System Objects:  3               Version:                1.5
Overflow Pages:       0               Overflow HighWater:     0
Thrashing Objects:    0               Thrashing Resolution:   0
Token Revocations:    0               Revocation Wait Time:   0.000
Devno:                0               Space Monitoring:       0,0
Quiescing System:     n/a             Quiescing Job Name:     n/a
Quiescor ASID:        n/a             File System Grow:       OFF,0
Status:               NM,EN,CO
Audit Fid:            D7F2E4E2 F1F60483 0000
Backups:              0               Backup File Space:      0K
File System Creation Time: May 21 12:17:39 2018
Time of Ownership:         May 21 12:18:14 2018
Statistics Reset Time:     May 21 12:18:14 2018
Quiesce Time:              n/a
Last Grow Time:            n/a
Connected Clients:    n/a

Legend: NM=Not mounted, EN=Encrypted, CO=Compressed
```

Looking at an IDCAMS LISTCAT listing from a z/OS V2R3 system, we see the **ZFS** and the **KEYLABEL** attributes associated with the dataset. The EXTENDED attribute is not associated with the dataset.

```
…
LISTCAT ENTRIES(OMVSSPT.ZFSEC.NONEXT.ZFS) -
ALL
CLUSTER ------- OMVSSPT.ZFSEC.NONEXT.ZFS
…
ENCRYPTIONDATA
DATA SET ENCRYPTION----(YES)
DATA SET KEY LABEL-----ZFS.AES.SECURE.KEY
…
SHROPTNS(3,3)   RECOVERY     UNIQUE           NOERASE     LINEAR
NONSPANNED           ZFS
…
```

File system activity:

After mounting the OMVSSPT.ZFSEC.ZFS file system on the /zfsecext mountpoint and having activity in the file system, we checked the files with the *zfsadm fileinfo* command.

Listing a directory (/zfsecext/Z1/D0) in the extended format file system, we see that there are some files from sizes of around 7K to 10K. Using the *zfsadm fileinfo* command, we can see the total bytes in kilobytes that were saved by the compress operation for each file and that they were encrypted. Note that the files which were 8K and under were determined to not have savings if compressed, however they remain eligible for compression.

Since we have enabled the userids for OMVS and zFS to access the encryption keys, we can access the files and directories via UNIX access methods (e.g. permissions, ACLs, security labels, FSACCESS, etc), even though the individual user is not permitted to the encryption key.

ls -al /zfsecext/Z1/D0

```
…
-rw-r--r--   1 bpxroot  sys1        7001 Jun  5 12:45 F2I
-rw-r--r--   1 bpxroot  sys1        8000 Jun  5 12:45 F2J
-rw-r--r--   1 bpxroot  sys1        9001 Jun  5 12:45 F2K
-rw-r--r--   1 bpxroot  sys1       10000 Jun  5 12:45 F2L
…
```

Here are some excerpts from the *zfsadm fileinfo* command for each file indicating encryption and compression.

/bin/zfsadm fileinfo -path /zfsecext/Z1/D0/F2I

```
…
encrypted                               compress-eligible 0K saved
```

/bin/zfsadm fileinfo -path /zfsecext/Z1/D0/F2J

```
…
encrypted                               compress-eligible 0K saved
```

/bin/zfsadm fileinfo -path /zfsecext/Z1/D0/F2K

```
…
encrypted                               compressed 8K saved
```

/bin/zfsadm fileinfo -path /zfsecext/Z1/D0/F2L

```
…
encrypted                               compressed 8K saved
```

The file system can be unencrypted and uncompressed using the *zfsadm decrypt* and *zfsadm decompress* commands. Here are the commands we used and excerpts from the *zfsadm fileinfo* and *zfsadm fsinfo* commands.

/bin/zfsadm decrypt -aggregate OMVSSPT.ZFSEC.ZFS

```
IOEZ00878I Aggregate OMVSSPT.ZFSEC.ZFS is successfully decrypted.
```

A message on the console will also be displayed.

```
*IOEZ00888I OMVSSPT.ZFSEC.ZFS is being encrypted or decrypted.
```

/bin/zfsadm fileinfo -path /zfsecext/Z1/D0/F2L

```
…
```

```
not encrypted                          compressed 8K saved
```

/bin/zfsadm decompress -aggregate OMVSSPT.ZFSEC.ZFS

```
IOEZ00900I Aggregate OMVSSPT.ZFSEC.ZFS is successfully decompressed.
```

A message on the console will also be displayed.

```
*IOEZ00898I OMVSSPT.ZFSEC.ZFS is being compressed or decompressed.
```

/bin/zfsadm fileinfo -path /zfsecext/Z1/D0/F2L

```
…
not encrypted                          not compressed
```

/bin/zfsadm fsinfo OMVSSPT.ZFSEC.ZFS -basic

```
OMVSSPT.ZFSEC.ZFS                       Z2       RW,RS,NE,NC

Legend: RW=Read-write, RS=Mounted RWSHARE, NE=Not encrypted
        NC=Not compressed
```

Note that after the file system has been decrypted there is still a key-label associated with the dataset and is protected from unauthorized access.

```
…
LISTCAT ENTRIES(OMVSSPT.ZFSEC.ZFS) -
ALL
CLUSTER ------- OMVSSPT.ZFSEC.ZFS
…
 ENCRYPTIONDATA
  DATA SET ENCRYPTION----(YES)
  DATA SET KEY LABEL-----ZFS.AES.SECURE.KEY
…
```

Compression and Encryption of existing zFS file systems.

Existing zFS file systems can be encrypted, compressed, or both. The dataset does not need to be SMS-managed extended format.

We compressed and encrypted the following existing and mounted zFS file system. This file system was originally a version 1.4 aggregate, consisting of v4 directories.

First, we made a backup of the file system before compressing and encrypting. In our environment, we use DFSMShsm to provide backup for the file systems.

Looking at the existing file system information we can see that it is at version 1.4.

/bin/zfsadm fsinfo omvsspt.legacy.zfs

```
File System Name: OMVSSPT.LEGACY.ZFS
*** owner information ***
Owner:               Z1             Converttov5:           OFF,n/a
Size:                4190400K       Free 8K Blocks:        358189
Free 1K Fragments:   42731          Log File Size:         23776K
```

```
Bitmap Size:          592K            Anode Table Size:       6296K
File System Objects: 24955            Version:                1.4
Overflow Pages:       0               Overflow HighWater:     0
Thrashing Objects:    0               Thrashing Resolution:   0
Token Revocations:    0               Revocation Wait Time:   0.000
Devno:                44758           Space Monitoring:       0,0
Quiescing System:     n/a             Quiescing Job Name:     n/a
Quiescor ASID:        n/a             File System Grow:       ON,0
Status:               RW,RS,NE,NC
Audit Fid:            D7F2E4E2 F0F618EF 0000
Backups:              0               Backup File Space:      0K

File System Creation Time: Mar 20 09:13:10 2006
Time of Ownership:         Jun 13 11:07:53 2018
Statistics Reset Time:     Jun 13 11:07:53 2018
Quiesce Time:              n/a
Last Grow Time:            n/a

Connected Clients:    Z3 Z2 Z4


Legend: RW=Read-write, RS=Mounted RWSHARE, NE=Not encrypted
NC=Not compressed
```

We tried compressing and encrypting this version 1.4 zFS aggregate and received reason code EF176C51, which indicated that encrypt, decrypt, compress, or decompress is not allowed for an aggregate version earlier than 1.5.

/bin/zfsadm compress -aggregate omvsspt.legacy.zfs

```
IOEZ00901E Error compressing aggregate OMVSSPT.LEGACY.ZFS, error code=121
reason code=EF176C51.
```

/bin/zfsadm encrypt -aggregate omvsspt.legacy.zfs

```
IOEZ00879E Error encrypting aggregate OMVSSPT.LEGACY.ZFS, error code=121
reason code=EF176C51.
```

We converted the aggregate to a version 1.5 aggregate, using *zfsadm convert*, keeping the v4 directories.

/bin/zfsadm convert -aggrversion omvsspt.legacy.zfs

```
IOEZ00810I Successfully changed aggregate omvsspt.legacy.zfs to version 1.5.
```

On the console we can see the following message.

```
IOEZ00650I Successfully changed the attribute of aggregate
OMVSSPT.LEGACY.ZFS.
```

Displaying the aggregate information, we see it is now a version 1.5 aggregate.

/bin/zfsadm fsinfo -aggregate omvsspt.legacy.zfs

```
File System Name: OMVSSPT.LEGACY.ZFS

*** owner information ***
Owner:                Z1              Converttov5:            OFF,n/a
```

```
Size:                 4190400K        Free 8K Blocks:          358189
Free 1K Fragments:    42731           Log File Size:           23776K
Bitmap Size:          592K            Anode Table Size:        6296K
File System Objects: 24955            Version:                 1.5
Overflow Pages:       0               Overflow HighWater:      0
Thrashing Objects:    0               Thrashing Resolution:    0
Token Revocations:    0               Revocation Wait Time:    0.000
Devno:                44758           Space Monitoring:        0,0
Quiescing System:     n/a             Quiescing Job Name:      n/a
Quiescor ASID:        n/a             File System Grow:        ON,0
Status:               RW,RS,NE,NC
Audit Fid:            D7F2E4E2 F0F618EF 0000
Backups:              0               Backup File Space:       0K

File System Creation Time: Mar 20 13:13:10 2006
Time of Ownership:         Jun 13 15:07:53 2018
Statistics Reset Time:     Jun 13 15:07:53 2018
Quiesce Time:              n/a
Last Grow Time:            n/a

Connected Clients:    Z3 Z2 Z4


Legend: RW=Read-write, RS=Mounted RWSHARE, NE=Not encrypted
NC=Not compressed
```

The command to compress the file system was then issued.

/bin/zfsadm compress -aggregate omvsspt.legacy.zfs

On the console we see the following message.

```
*IOEZ00898I OMVSSPT.LEGACY.ZFS is being compressed or decompressed.
```

Displaying the file system information with *zfsadm fsinfo*, we can monitor the percent completion. Also, the *'modify omvs,pfs=zfs,fsinfo…'* console command can be used. During the operation the display will show a partial completion.

/bin/zfsadm fsinfo omvsspt.legacy.zfs

```
File System Name: OMVSSPT.LEGACY.ZFS

*** owner information ***
Owner:                Z1              Converttov5:             OFF,n/a
Size:                 4190400K        Free 8K Blocks:          368160
Free 1K Fragments:    42731           Log File Size:           23776K
Bitmap Size:          592K            Anode Table Size:        6296K
File System Objects: 24955            Version:                 1.5
Overflow Pages:       0               Overflow HighWater:      0
Thrashing Objects:    0               Thrashing Resolution:    0
Token Revocations:    0               Revocation Wait Time:    0.000
Devno:                44758           Space Monitoring:        0,0
Quiescing System:     n/a             Quiescing Job Name:      n/a
Quiescor ASID:        n/a             File System Grow:        ON,0
Status:               RW,RS,NE,CI
Audit Fid:            D7F2E4E2 F0F618EF 0000
Backups:              0               Backup File Space:       0K
```

```
Compress Progress:   running, 23% complete started at Jun 13 12:08:46 2018
task 6AA528

File System Creation Time: Mar 20 09:13:10 2006
Time of Ownership:         Jun 13 11:07:53 2018
Statistics Reset Time:     Jun 13 11:07:53 2018
Quiesce Time:              Jun 13 12:08:46 2018
Last Grow Time:            n/a

Connected Clients:   Z3 Z2 Z4


Legend: RW=Read-write, RS=Mounted RWSHARE, NE=Not encrypted
CI=Partially compressed
```

When the compression is complete we see the IOEZ00899I message.

```
IOEZ00899I Aggregate OMVSSPT.LEGACY.ZFS is successfully compressed.
```

Displaying the file system information with *zfsadm fsinfo*, we now see that the file system is compressed.

/bin/zfsadm fsinfo omvsspt.legacy.zfs

```
File System Name: OMVSSPT.LEGACY.ZFS

*** owner information ***
Owner:                Z1              Converttov5:            OFF,n/a
Size:                 4190400K        Free 8K Blocks:         469275
Free 1K Fragments:    42731           Log File Size:          23776K
Bitmap Size:          592K            Anode Table Size:       6296K
File System Objects:  24955           Version:                1.5
Overflow Pages:       0               Overflow HighWater:     0
Thrashing Objects:    0               Thrashing Resolution:   0
Token Revocations:    0               Revocation Wait Time:   0.000
Devno:                44758           Space Monitoring:       0,0
Quiescing System:     n/a             Quiescing Job Name:     n/a
Quiescor ASID:        n/a             File System Grow:       ON,0
Status:               RW,RS,NE,CO
Audit Fid:            D7F2E4E2 F0F618EF 0000
Backups:              0               Backup File Space:      0K

File System Creation Time: Mar 20 09:13:10 2006
Time of Ownership:         Jun 13 11:07:53 2018
Statistics Reset Time:     Jun 13 11:07:53 2018
Quiesce Time:              n/a
Last Grow Time:            n/a

Connected Clients:   Z3 Z2 Z4


Legend: RW=Read-write, RS=Mounted RWSHARE, NE=Not encrypted, CO=Compressed
```

Looking at a couple of files using the *zfsadm fileinfo* command, we can see the percent saved. For the file that is 8K or under, it remains eligible, but was not compressed.

Ls -al /legacy/JUNK.WK4

```
-rwxr-xr-x    1 bpxroot  sys1     1399724 Feb  7  2002 JUNK.WK4

/bin/zfsadm fileinfo -path /legacyzfs/JUNK.WK4
path: /legacyzfs/JUNK.WK4
***   global data   ***
fid                     3,3           anode                   177922,1020
length                  1399724       format                  BLOCKED
1K blocks               512           permissions             755
uid,gid                 0,0           access acl              0,0
dir model acl           na            file model acl          na
user audit              F,F,F         auditor audit           N,N,N
set sticky,uid,gid      0,0,0         seclabel                none
object type             FILE          object linkcount        1
object genvalue         0             dir version             na
dir name count          na            dir data version        na
dir tree status         na            dir conversion          na
file format bits        0x0,0,0       file charset id         0x0
file cver               none          charspec major,minor    na
direct blocks           0x00048842    0x00048843    0x00048844    0x80000803
0x80000000    0x80000000    0x80000000    0x80000000
indirect blocks         0x00008D6E
mtime       Feb  7 18:24:19 2002     atime        Aug  5 09:42:02 2017
ctime       Mar 20 13:13:32 2006     create time  Mar 20 13:13:32 2006
reftime     none
not encrypted                        compressed 864K saved
```

ls -al /legacyzfs/U027001/.sh_history

```
-rw-------   1 U027001  sys1 97 Jan 25  2011 /legacyzfs/U027001/.sh_history
```

/bin/zfsadm fileinfo /legacyzfs/U027001/.sh_history

```
path: /legacyzfs/U027001/.sh_history
***   global data   ***
fid                     24961,1922644 anode                   46507,516
length                  97            format                  FRAGMENTED
1K blocks               1             permissions             600
uid,gid                 3879,0        access acl              0,0
dir model acl           na            file model acl          na
user audit              F,F,F         auditor audit           N,N,N
set sticky,uid,gid      0,0,0         seclabel                none
object type             FILE          object linkcount        1
object genvalue         0             dir version             na
dir name count          na            dir data version        na
dir tree status         na            dir conversion          na
file format bits        0x0,0,0       file charset id         0x0
file cver               none          charspec major,minor    na
fragment location       224704        6             1
indirect blocks         none
mtime       Jan 26 03:00:34 2011     atime        Aug  5 09:42:46 2017
ctime       Jan 26 03:00:34 2011     create time  Jan 26 02:59:46 2011
reftime     none
not encrypted                        compress-eligible 0K saved
```

ls -al /legacyzfs/U040001/data

```
-rwxrwxr-x   1 bpxroot  sys1     1245970 May 22  2002 /legacyzfs/U040001/data
```

/bin/zfsadm fileinfo -path /legacyzfs/U040001/data

```
path: /legacyzfs/U040001/data
***   global data   ***
fid                   17377,17377   anode                 27393,516
length                1245970       format                BLOCKED
1K blocks             320           permissions           775
uid,gid               0,0           access acl            0,0
dir model acl         na            file model acl        na
user audit            F,F,F         auditor audit         N,N,N
set sticky,uid,gid    0,0,0         seclabel              none
object type           FILE          object linkcount      1
object genvalue       0             dir version           na
dir name count        na            dir data version      na
dir tree status       na            dir conversion        na
file format bits      0x0,0,0       file charset id       0x0
file cver             none          charspec major,minor  na
direct blocks         0x0004F210    0x0004F211    0x80000802    0x80000000
0x80000000    0x80000000    0x80000000    0x80000000
indirect blocks       0x00022E6B
mtime       May 22 13:43:28 2002    atime        Sep 11 20:28:51 2017
ctime       Oct 13 15:38:06 2009    create time  Mar 20 09:20:07 2006
reftime     none
not encrypted                       compressed 912K saved
```

We then issued the command to encrypt the file system, without the key label and received the reason code EF176C6F, which indicates that the encryption operation cannot continue without a key label.

/bin/zfsadm encrypt -aggregate omvsspt.legacy.zfs

```
IOEZ00879E Error encrypting aggregate OMVSSPT.LEGACY.ZFS, error code=121
reason code=EF176C6F.
```

Next, we issued the encrypt command with a key label provided by our ICSF administrator.

/bin/zfsadm encrypt -aggregate omvsspt.legacy.zfs -keylabel ZFS.AES.SECURE.KEY

On the console we see the following message.

```
*IOEZ00888I OMVSSPT.LEGACY.ZFS is being encrypted or decrypted.
```

Displaying the file system with *zfsadm fsinfo*, we can see the percentage complete for the encryption operation, and that it is partially encrypted. Notice that it is still fully compressed.

/bin/zfsadm fsinfo omvsspt.legacy.zfs

```
File System Name: OMVSSPT.LEGACY.ZFS

*** owner information ***
Owner:               Z1             Converttov5:           OFF,n/a
Size:                4190400K       Free 8K Blocks:        467680
Free 1K Fragments:   33647          Log File Size:         23776K
Bitmap Size:         592K           Anode Table Size:      6296K
File System Objects: 24955          Version:               1.5
```

```
Overflow Pages:      0               Overflow HighWater:     0
Thrashing Objects:   0               Thrashing Resolution:   0
Token Revocations:   0               Revocation Wait Time:   0.000
Devno:               44758           Space Monitoring:       0,0
Quiescing System:    n/a             Quiescing Job Name:     n/a
Quiescor ASID:       n/a             File System Grow:       ON,0
Status:              RW,RS,EI,CO
Audit Fid:           D7F2E4E2 F0F618EF 0000
Backups:             0               Backup File Space:      0K
Encrypt Progress:    running, 24% complete started at Jun 13 15:05:01 2018
task 6AA078

File System Creation Time: Mar 20 09:13:10 2006
Time of Ownership:         Jun 13 11:07:53 2018
Statistics Reset Time:     Jun 13 11:07:53 2018
Quiesce Time:              n/a
Last Grow Time:            n/a

Connected Clients:   Z3 Z2 Z4


Legend: RW=Read-write, RS=Mounted RWSHARE, EI=Partially encrypted
CO=Compressed
```

When the encryption is complete we see the following message.

```
IOEZ00877I Aggregate OMVSSPT.LEGACY.ZFS is successfully encrypted.
```

Displaying the file system information, we see that the file system is now fully compressed and encrypted.

/bin/zfsadm fsinfo omvsspt.legacy.zfs

```
File System Name: OMVSSPT.LEGACY.ZFS

*** owner information ***
Owner:               Z1              Converttov5:            OFF,n/a
Size:                4190400K        Free 8K Blocks:         465611
Free 1K Fragments:   0               Log File Size:          23776K
Bitmap Size:         592K            Anode Table Size:       6296K
File System Objects: 24955           Version:                1.5
Overflow Pages:      0               Overflow HighWater:     0
Thrashing Objects:   0               Thrashing Resolution:   0
Token Revocations:   0               Revocation Wait Time:   0.000
Devno:               44758           Space Monitoring:       0,0
Quiescing System:    n/a             Quiescing Job Name:     n/a
Quiescor ASID:       n/a             File System Grow:       ON,0
Status:              RW,RS,EN,CO
Audit Fid:           D7F2E4E2 F0F618EF 0000
Backups:             0               Backup File Space:      0K

File System Creation Time: Mar 20 09:13:10 2006
Time of Ownership:         Jun 13 11:07:53 2018
Statistics Reset Time:     Jun 13 11:07:53 2018
Quiesce Time:              n/a
Last Grow Time:            n/a

Connected Clients:   Z3 Z2 Z4
```

```
Legend: RW=Read-write, RS=Mounted RWSHARE, EN=Encrypted, CO=Compressed
```

Using IDCAMS LISTCAT, we see the key label and the ZFS attribute associated with the dataset.

```
…
LISTCAT ENTRIES(OMVSSPT.LEGACY.ZFS) -
ALL
CLUSTER ------- OMVSSPT.LEGACY.ZFS
...
ENCRYPTIONDATA
  DATA SET ENCRYPTION----(YES)
  DATA SET KEY LABEL-----ZFS.AES.SECURE.KEY
...
SHROPTNS(3,3)   RECOVERY     UNIQUE           NOERASE     LINEAR
NONSPANNED           ZFS
...
```

Displaying some files, we see that they are compressed and encrypted.

ls -al /legacyzfs/JUNK.WK4

```
-rwxr-xr-x   1 bpxroot  sys1     1399724 Feb  7  2002 /legacyzfs/JUNK.WK4
```

/bin/zfsadm fileinfo -path /legacyzfs/JUNK.WK4

```
path: /legacyzfs/JUNK.WK4
***   global data   ***
fid                     3,3           anode                   177922,1020
length                  1399724       format                  BLOCKED
1K blocks               512           permissions             755
uid,gid                 0,0           access acl              0,0
dir model acl           na            file model acl          na
user audit              F,F,F         auditor audit           N,N,N
set sticky,uid,gid      0,0,0         seclabel                none
object type             FILE          object linkcount        1
object genvalue         0             dir version             na
dir name count          na            dir data version        na
dir tree status         na            dir conversion          na
file format bits        0x0,0,0       file charset id         0x0
file cver               none          charspec major,minor    na
direct blocks           0x0005166E    0x0005166F    0x00051670    0x80000803
0x80000000    0x80000000    0x80000000    0x80000000
indirect blocks         0x00008D6E
mtime       Feb  7 13:24:19 2002     atime        Aug  5 05:42:02 2017
ctime       Mar 20 09:13:32 2006     create time  Mar 20 09:13:32 2006
reftime     none
encrypted                            compressed 864K saved
```

ls -al /legacyzfs/U027001/.sh_history

```
-rw-------   1 U027001  sys1 97 Jan 25  2011 /legacyzfs/U027001/.sh_history
```

/bin/zfsadm fileinfo /legacyzfs/U027001/.sh_history

```
path: /legacyzfs/U027001/.sh_history
***   global data   ***
```

```
fid                     24961,1922644 anode                   46507,516
length                  97            format                  BLOCKED
1K blocks               8             permissions             600
uid,gid                 3879,0        access acl              0,0
dir model acl           na            file model acl          na
user audit              F,F,F         auditor audit           N,N,N
set sticky,uid,gid      0,0,0         seclabel                none
object type             FILE          object linkcount        1
object genvalue         0             dir version             na
dir name count          na            dir data version        na
dir tree status         na            dir conversion          na
file format bits        0x0,0,0       file charset id         0x0
file cver               none          charspec major,minor    na
direct blocks           0x0005D816
indirect blocks         none
mtime       Jan 25 22:00:34 2011      atime        Aug  5 05:42:46 2017
ctime       Jan 25 22:00:34 2011      create time  Jan 25 21:59:46 2011
reftime     none
encrypted                             compress-eligible OK saved
```

ls -al /legacyzfs/U040001/data

```
-rwxrwxr-x   1 bpxroot  sys1     1245970 May 22  2002 /legacyzfs/U040001/data
```

/bin/zfsadm fileinfo -path /legacyzfs/U040001/data

```
path: /legacyzfs/U040001/data
***   global data   ***
fid                     17377,17377   anode                   27393,516
length                  1245970       format                  BLOCKED
1K blocks               320           permissions             775
uid,gid                 0,0           access acl              0,0
dir model acl           na            file model acl          na
user audit              F,F,F         auditor audit           N,N,N
set sticky,uid,gid      0,0,0         seclabel                none
object type             FILE          object linkcount        1
object genvalue         0             dir version             na
dir name count          na            dir data version        na
dir tree status         na            dir conversion          na
file format bits        0x0,0,0       file charset id         0x0
file cver               none          charspec major,minor    na
direct blocks           0x0005A917    0x0005A918    0x80000802    0x80000000
0x80000000    0x80000000    0x80000000    0x80000000
indirect blocks         0x00022E6B
mtime       May 22 13:43:28 2002      atime        Sep 11 20:28:51 2017
ctime       Oct 13 15:38:06 2009      create time  Mar 20 09:20:07 2006
reftime     none
encrypted                             compressed 912K saved
```

ls -al /legacyzfs/U078002/.sh_history

```
-rw-------  1 bpxroot  sys1 196698 May 8 2002 /legacyzfs/U078002/.sh_history
```

/bin/zfsadm fileinfo -path /legacyzfs/U078002/.sh_history

```
path: /legacyzfs/U078002/.sh_history
***   global data   ***
fid                     803,803       anode                   18763,1020
length                  196698        format                  BLOCKED
```

```
1K blocks              40            permissions            600
uid,gid                0,0           access acl             0,0
dir model acl          na            file model acl         na
user audit             F,F,F         auditor audit          N,N,N
set sticky,uid,gid     0,0,0         seclabel               none
object type            FILE          object linkcount       1
object genvalue        0             dir version            na
dir name count         na            dir data version       na
dir tree status        na            dir conversion         na
file format bits       0x0,0,0       file charset id        0x0
file cver              none          charspec major,minor   na
direct blocks          0x000518FB    0x80000801    0x80000000    0x80000000
0x80000000    0x80000000    0x80000000    0x80000000
indirect blocks        0x0002A9DF
mtime       May  8 19:09:06 2002     atime        Sep 11 20:31:47 2017
ctime       Mar 20 09:13:50 2006     create time  Mar 20 09:13:50 2006
reftime     none
encrypted                            compressed 168K saved
```

Using the zFS online salvage command, we can verify that the file system, which is now encrypted and compressed, is verified with no errors.

/bin/zfsadm salvage OMVSSPT.LEGACY.ZFS

```
IOEZ00711I Aggregate OMVSSPT.LEGACY.ZFS successfully verified or repaired.
```

On the console we can see the following messages.

IOEZ00729I Verification of aggregate OMVSSPT.LEGACY.ZFS started
IOEZ00705I Formatted v5 aggregate size 523800 8K blocks, dataset size 523800 8K blocks
IOEZ00707I Log file size 2969 8K blocks, verified correct
IOEZ00709I Bitmap size 73 8K blocks, verified correct
IOEZ00951I Aggregate OMVSSPT.LEGACY.ZFS anode table length=2333(in 8K blocks) LPI=0 encrypted compressed
IOEZ00782I Salvage has verified 78 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 234 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 312 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 390 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 468 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 546 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 702 of 784 pages in the anode table.
IOEZ00782I Salvage has verified 281 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 292 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 562 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 584 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 876 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 843 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 1168 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 1124 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 1460 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 1405 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 1752 of 2922 directory block in the directory tree.

IOEZ00782I Salvage has verified 1686 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 2044 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 1967 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 2336 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 2248 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 2628 of 2922 directory block in the directory tree.
IOEZ00782I Salvage has verified 2529 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 2818 of 2818 directories in the directory tree.
IOEZ00782I Salvage has verified 5 of 5 pages in the partially-free page list.
IOEZ00782I Salvage has verified 2 of 2 pages in the totally free page stack.
IOEZ00722I Primary file system size 2333 8K blocks, verified correct
IOEZ00739I Salvage processed 2921 directory pages, 24956 anodes, 1497 indirect blocks and 784 anode table pages.
IOEZ00730I Verification of aggregate OMVSSPT.LEGACY.ZFS completed, no errors found.